Introduction to Scripting

Canterbury Linux Users Group

Carey Evans

10 August 1998

Introduction

- ☞ What?
- ☞ Why?
- ☞ Which?
- ☞ How?
- ☞ Who?
- ☞ Where?

What is a Script?

What is a script?

- ☞ Short program
- ☞ Not compiled
- ☞ Buzzword compliant:
  - ✓ (Very) High Level Language
  - ✓ Rapid Application Development (RAD)

What are scripts used for?

☞ Automating tedious, complex or repetitive tasks

☞ Text processing

☞ Saving ten minutes work by spending an hour writing a script to do it for you

Why script?

Why use a script, instead of C, C++, etc.?

☞ Shallower learning curve

☞ Faster development

☞ Easier debugging

☞ Better string, list and hash support

Why learn to script at all?

- ☞ Make things easier for yourself
- ☞ Customise your environment
- ☞ Impress friends/co-workers/boss
- ☞ Wake up one day and discover you're a computer programmer

Which scripting languages exist?

From tools to real languages

- ☞ POSIX /bin/sh, textutils, shellutils, sed and awk
- ☞ Tcl/Tk
- ☞ Perl
- ☞ Python

Common features

✓ String data type and operations, including regular expressions

✓ Arithmetic operations

✓ Powerful control structures, including user defined functions

✓ Extensible

✓ Free with every modern Linux distribution!

Other embedded languages

☞ Emacs Lisp

☞ Guile — GIMP and Gnome

☞ MUD and IRC scripting languages

☞ Java applets and Javascript — WWW browsers

Not covered in this introduction.

Other platforms

- DOS batch files — aren't very good at *anything*
- REXX — Part of OS/2, Amiga and OS/400, available for PC-DOS and Unix
- AppleScript, DCL (VMS), CL (OS/400)
- LotusScript (Notes/Domino), VBA (MS Office)
- Many, many others...

How do they compare?

/bin/sh

✓ A POSIX shell is quite powerful, especially for things involving other programs, and it should be available on any Unix. There are lots of tools available. The install process for Debian 1.3 was written for ash.

✗ However, larger scripts get hard to maintain, and only simple data types are available. The quoting of strings containing whitespace is not intuitive. It's not very fast. There is very limited graphics.

Tcl/Tk

✓ The GUI integration (with Tk) is very good. Everything is a string. It's possible to extend Tcl with C, or embed Tcl in C, without too much trouble.

✗ Everything is a string. Quoting is bizarre. Larger scripts can become hard to maintain. It was slow, but this is improving.

Perl

✓ *Excellent* text processing support. Lots of contributed modules. Very widely used. Object orientation, module and name-space support. Supports Tk. It's possible to embed Perl in C (e.g. Apache) or extend Perl with C (many of the modules). There's more than one way to do it.

✗ Overuse of punctuation. Many traps for the unwary. There's more than one way to do it.

Python

✓ More design than accumulation. Control flow is based on indentation, and punctuation is used sparingly, making Python programs look cleaner. Real object orientation. Scalable to large projects. Supports Tk, and Perl regular expressions. Can be embedded and extended. Good use of exceptions.

✗ Control flow is based on indentation, which can be strange for new users. Reference based variables can have unexpected consequences.

Who wants to see some examples?

Simple examples

- Ask someone their name and if it's their birthday. If it is, 'sing' happy birthday to them.

- Sort the lines in a text file, where the fields are separated by colons, by the number in the third field, and output the lines where this number is 1000 or greater.

Less simple examples

- Ask someone their name and if it’s their birthday, but this time do it graphically.

- Check whether a web page has been updated.

Where can I find out more?

Home pages

Python: `http://www.python.org/`

Perl: `http://language.perl.com/`

Tcl: `http://www.scriptics.com/resource/`

Newsgroups

Python: `comp.lang.python`

Perl: `comp.lang.perl.*`

Tcl: `comp.lang.tcl.*`

Colophon

These slides were produced with pdfTEX from teTEX 0.9 on Debian GNU/Linux 2.0, using the slides document class, with the help of *The LATEX Companion* and AUC-TEX.

References included *Programming Perl*, *Programming Python*, many Info files and man pages, the Perl and Python language comparisons on their respective web sites, and a variety of other web pages.

The slides were written with the help of Garbage, Catatonia, Meredith Brooks, Savage Garden, Republica, and some excellent built-in monitor speakers.